# An introduction to *Macaulay2* packages

Mike Stillman (mike@math.cornell.edu)

Department of Mathematics
Cornell

2 June 2020 / Warwick Macaulay2 workshop

# What is a *Macaulay2* package?

**_Macaulay2 Package_**

A file (or a file and a folder) containing Macaulay2 code for doing a specific task (or tasks) and also containing documentation and tests for its functionality.

Using packages:

- Many packages are distributed with Macaulay2.
- Load package: `needsPackage "Foo"` or `loadPackage "Foo"` (loads the file `"Foo.m2"`)
- Create documentation: `installPackage "Foo"` (including Macaulay2 example inputs and outputs).
- View documentation: `viewHelp "Foo"`
- Run tests: `check Foo`
- Also useful: `uninstallPackage "Foo"`

# Why should *I* write one?

Write a package for your future self, or others. I start a package for most every mathematical project I work on.

## Reasons

- organizes your hard programming work
- documents your functions and methods (lets you remember what is in your package!)
- tests, for confidence that your code is still working.
- The tests also help improve your interface so your package is easy to use.
- If you distribute it with M2: allows us to make sure it keeps working with future versions of Macaulay2.

- have it included in Macaulay2 (issue a pull request at `https://github.com/Macaulay2/M2`).
- submit to Journal of Software in Algebra and Geometry (`https://msp.org/jsag`).

# What are the parts of a package?

The parts of your package:

- The `newPackage` preamble
- The `export` section.
- Your code!
- The `beginDocumentation` section.
  - Documentation for each function, method, and symbol you export.
  - Tests: make sure your functions and methods do what you think they should.

# Let's write a package

Let's do an example!

We will use: `packageTemplate "Foo"` to get us started. We will write a package that computes a Hilbert function for an ideal, with less typing.

See

**Resources to help**

- `packageTemplate "Foo"`
- Look at some well-written packages, e.g. `NormalToricVarieties`.
- Documentation: "creating a package", Package, TEST, assert, doc, SimpleDoc.
- `https://github.com/Macaulay2/M2/wiki/Package-Writing-Style-Guide`

# An exercise!

**Exercise**

Start with the online file `BuggyPackage.m2` in `NewToM2`. Fix the bugs in it. Then rename the package to something else (BuggyNoMore?), and load that.

# Some frequently asked questions

- I get the error message `error: mutable unexported unset symbol(s) in package MyExcellentPackage: 'R', 't'` How do I fix this?
- What is DebuggingMode?
- How do I break up my package into more files?
- I make a change to my package, but when I load it, it doesn't seem to recognize it?

To summarize: Write a package for your future self, or others.

- organizes your hard programming work
- documents your functions and methods
- tests, for confidence that your code is still working.

**Thanks!**

To summarize: Write a package for your future self, or others.

- organizes your hard programming work
- documents your functions and methods
- tests, for confidence that your code is still working.

**Thanks!**